

All or Nothing Caching Games with Bounded Queries

Dömötör Pálvölgyi*

February 3, 2017

Abstract

We determine the value of some search games where our goal is to find all of some hidden treasures using queries of bounded size. The answer to a query is either empty, in which case we lose, or a location, which contains a treasure. We prove that if we need to find d treasures at n possible locations with queries of size at most k , then our chance of winning is $\frac{k^d}{\binom{n}{d}}$ if each treasure is at a different location and $\frac{k^d}{\binom{n+d-1}{d}}$ if each location might hide several treasures for large enough n . Our work builds on some results by Csóka who has studied a continuous version of this problem, known as Alpern's Caching Game; we also prove that the value of Alpern's Caching Game is $\frac{k^d}{\binom{n+d-1}{d}}$ for integer k and large enough n .

1 Introduction

We consider the following game theoretic search problem. Initially a player, called *hider*, hides d treasures behind n doors and after the hiding is over, another player, *searcher*, needs to find them all. In each round searcher selects (guesses) at most k doors. If none of them has a treasure, she[†] loses. Otherwise, hider reveals one of the treasures behind one of the selected doors. If she finds all d treasures with a total of d guesses, she wins.

The value of the game is the chance of searcher winning if both players play optimally. If hider is allowed to hide more treasures behind the same door (Multiple Caching Game), we denote this value by $v_M(n, d, k)$, and if hider can hide at most one treasure behind each door (Single Caching Game), we denote this value by $v_1(n, d, k)$. Note that $v_M(n, d, k)$ decreases in n and in d and increases in k , while $v_1(n, d, k)$ is not monotone in d . Our main result is to determine both values if n is large enough.

Theorem 1. *The value of the Single Caching Game is $v_1(n, d, k) = \frac{k^d}{\binom{n}{d}}$ if $n \geq dk$.*

Theorem 2. *The value of the Multiple Caching Game is $v_M(n, d, k) = \frac{k^d}{\binom{n+d-1}{d}}$ if n is large enough (compared to k and d).*

The upper bounds follow from a simple counting argument and can be formulated in the following meta-lemma, whose continuous equivalent for some games was stated by Csóka [4].

*Department of Pure Mathematics and Mathematical Statistics, University of Cambridge. Research supported by the Marie Skłodowska-Curie action of the EU, under grant IF 660400.

[†]For simplicity, we use *she* for searcher and *he* for hider.

Lemma 3. *The value of any finite search game where searcher needs to find all hidden treasures is bounded from above by the maximal number of hiding possibilities a deterministic strategy of the searcher can reveal divided by the total number of hiding possibilities.*

Proof. If the treasures are hidden according to a fixed distribution, then it is enough to consider deterministic strategies of the searcher, as finite search games have a value.* The performance of these is bounded from above with the claimed quantity against uniformly choosing from the available hiding possibilities. \square

The proof of our lower bounds also builds on results by Csóka [4], who has studied a continuous version of the Multiple Caching Game, known as Alpern’s Caching Game [1]; see also [2]. In this game hider is allowed to dig holes at n possible places in a total of 1 unit depth, and then hide the d treasures while burying the holes back at any depth. For example, he can dig one hole 0.8 deep and another hole 0.2 deep, then hide a treasure in the first hole at depths 0.7 and another treasure in the same hole at depth 0.5, then a third treasure in the second hole at depth 0.2. After the hiding of the treasures is over, searcher can dig out a total of k depth and needs to find all d treasures. (She notices when she finds a treasure.)

Our motivation to introduce the Multiple Caching Game, which was to some extent already done implicitly in [4], was the following connection to Alpern’s Caching Game.

Observation 4. *The value of Alpern’s Caching Game is at least the value of the Multiple Caching Game if k is an integer, i.e., $v_A(n, d, k) \geq v_M(n, d, k)$.*

Proof. If k is an integer, then searcher can pretend that she is playing the Multiple Caching Game instead of Alpern’s Caching Game by always digging simultaneously in k holes with the same speed. In case there are treasures in more than one hole, in Alpern’s Caching Game the depth determines which she finds first; this is better for her than the rule of the Multiple Caching Game, according to which hider decides which treasure she finds. If she manages to find all k treasures, then she has to dig for at most k depth in total. Therefore if she wins the Multiple Caching Game, she also wins Alpern’s Caching Game with this strategy. \square

From Observation 4 and the lower bound of Theorem 2, combined with the upper bound given by (Csóka’s continuous version of) Lemma 3, we get the following.

Theorem 5. *The value of Alpern’s Caching Game is $v_A(n, d, k) = \frac{k^d}{\binom{n+d-1}{d}}$ if k is an integer and n is large enough (compared to k and d).*

Remark 6. Observation 4 would also hold for the following variant of the Multiple Caching Game. At the beginning we start with $P = k$ power and in each turn we can guess at most P doors. If we guess $\ell < P$ doors and none of them contains a treasure, then instead of losing immediately, we can continue the game with $P := P - \ell$ power from now on. If we denote the value of this new variant by v_P , then from the proof of Observation 4 we get $v_A(n, d, k) \geq v_P(n, d, k) \geq v_M(n, d, k)$ and from Theorem 5 that all values are equal for large n ’s.

Remark 7. We have introduced the rule that if there are treasures behind more than one guessed door, it is hider who reveals one of them, so that the argument used in the proof of Observation 4

*This holds even if the outcomes of the queries depend on any, possibly random parameter that is independent of the searcher’s strategy.

works. The game where we use instead the rule that a treasure is revealed from such doors randomly, chosen uniformly either according to the number of doors or to the number of treasures they hide, we get other natural variants of the problem. The values of these games are bounded from below by the value of the Multiple Caching Game, v_M , and the upper bounds given by Lemma 3 are also identical, thus the values of these random variants can only differ from v_M for small n 's.

The organization of the rest of the paper is as follows. In Section 2 we give some simple examples and give the (quite elementary) proof of Theorem 1. In Section 3 we prove our main result, Theorem 2. Section 4 discusses what happens for small values of n and Section 5 contains some further remarks and open problems. We end the Introduction with a very brief summary of previous results.

1.1 History of Alpern's Caching Game

The value of Alpern's Caching Game, $v_A(n, d, k)$, was determined for some small values in [1, 4, 5]. For example, it is easy to see that $v_A(n, 1, k) = \frac{\lfloor k \rfloor}{n}$, but the problem is open in general already for $d = 2$. Csóka [4] has shown that the uniform hiding strategy guarantees $v_A(n, d, k) \leq \frac{k^d}{\binom{n+d-1}{d}}$ using a continuous version of Lemma 3. He has also proved that $v_A(n, 2, k) = \frac{k^2}{\binom{n+1}{2}}$ whenever k is an integer and made the following two conjectures.

Conjecture 8 (Csóka [4]). $v_A(n, d, k) = \frac{k^d}{\binom{n+d-1}{d}}$ if k is an integer and $n \geq dk$.

Conjecture 9 (Csóka [4]). $v_A(n, d, k) = \frac{k^d}{\binom{n+d-1}{d}}$ if $k \geq 1 + \frac{2}{d-1}$ and $n \geq dk$.

Notice that Conjecture 9 is stronger than Conjecture 8 (discounting some small, known cases). Csóka gave a construction that shows why $k \geq 1 + \frac{2}{d-1}$ is needed and suggested a searching strategy type that requires $n \geq dk$; our strategy for searcher also falls in this category. This searching strategy type was inspired by another interesting variant of Alpern's Caching Game, where instead of n doors the hider can hide the treasures anywhere under a measurable interval of length n ; this version was introduced and solved almost completely by Csóka [4], proving that its value equals $\frac{k^d}{\binom{n+d-1}{d}}$ if $n \geq dk$.

Theorem 5 solves Conjecture 8 if n is large enough. We conjecture that in fact Conjecture 8 and Theorem 2 already hold for $n = dk - 1$. For non-integer k our results have no implication and thus Conjecture 9 remains wide open, possibly also true already for all $n \geq dk - 1$. This slightly lower bound was conjectured by Csóka for $d = 2$, but the answer is known for no $3 < k \notin \mathbb{Z}$.

2 Warm-up and Single Caching Game

As a warm-up, let us consider the Single Caching Game where searcher needs to find $d = 2$ treasures in $n = 4$ doors by guessing $k = 2$ doors in each round. After an initial guess of two doors, say, doors 1 and 2, she either loses immediately, or finds a treasure behind one of the doors, say, door 1. The other treasure can be behind doors 2, 3 or 4. Searcher has two options for her second (and final) guess: she can either guess the two doors that she hasn't guessed before, doors 3 and 4, or guess door 2 and one of 3 or 4. What should she do?

This situation is very similar to popular paradoxes, such as Bertrand's box paradox, the Three Prisoners problem and especially to the Monty Hall problem. In the latter problem, known from

a TV show hosted by Monty Hall, contestants were presented three doors, one of which hid a car and the other two goats. Their goal was to open the door that hid the car. The contestant could pick a door first (without opening it). Then the host has opened one of the other two doors, one that hid a goat. Finally, the contestant could decide whether to open the door she has originally picked, or the other unopened door. Many contestants opened their originally picked door, giving them a chance of $\frac{1}{3}$ to walk away with the car, opposed to $\frac{2}{3}$ had they decided to switch.

For illustration, let us calculate the chances of winning in the Single Caching Game for searcher when the treasures are distributed uniformly at random, i.e., each of the $\binom{4}{2}$ options has $\frac{1}{6}$ probability. If in the first round she picks doors 1 and 2, and in the second round doors 3 and 4, then she wins in 4 out of the 6 cases, i.e., with probability $\frac{2}{3}$. If in the first round she picks doors 1 and 2, and in the second round doors 2 and 3 (supposing that she has found a treasure behind door 1), then she wins in 3 out of the 6 cases, i.e., with probability $\frac{1}{2}$ (assuming that hider reveals both treasures with 50% chance if searcher guesses both with her first guess). This shows that in the Single Caching Game it is also better to switch.

Proof of Theorem 1. Though the upper bound follows from Lemma 3, we present it here for this special version in more detail. In this game hider doesn't have much options, the best he can do is to hide the d treasures uniformly at random. We need to prove that searcher has at most $\frac{k^d}{\binom{n}{d}}$ probability of winning against this distribution of the treasures. This is enough to prove for deterministic strategies of the searcher. Suppose that she picks doors X_i in round i (where X_i might depend on the answer to all X_j for $j < i$). To win, she needs each of the X_i to contain a treasure. (But each X_i containing a treasure doesn't imply that she wins!) There are $2^{|X_i|}$ possibilities for the outcomes. As in total there are $\binom{n}{d}$ possibilities, her chance of winning is at most $\frac{k^d}{\binom{n}{d}}$ against the uniform distribution. This can also be attained if she always picks k new doors at random, which can be done if $n \geq dk$, proving the lower bound. \square

3 Multiple Caching Game

This section contains the proof of our main result about the Multiple Caching Game.

Proof of Theorem 2. The upper bound follows from Lemma 3; searcher can cover at most k^d of the $\binom{n+d-1}{d}$ possibilities, and thus her chance of winning is at most $\frac{k^d}{\binom{n+d-1}{d}}$. (Alternatively, it also follows from Csóka's upper bound for Alpern's Caching Game and Observation 4.)

The lower bound is more involved. First, notice that if $k = 1$, then $v_M(n, d, 1) \geq \frac{1}{\binom{n+d-1}{d}}$ can be achieved with the following strategy: Searcher initially picks an allocation of the treasures, μ , uniformly at random from the $\binom{n+d-1}{d}$ possibilities, then in each round she selects a door with the aim of finding the treasures where they are in μ .

We will use this strategy for $k = 1$ to give a strategy for $k > 1$. To this end, suppose that in the above strategy for $k = 1$, searcher additionally follows the below rules.

- If she guesses a door which hides more treasures according to μ , then she keeps on guessing the same door in the subsequent rounds until she finds all the treasures behind this door.
- When she needs to guess a new door, she always guesses the one which hides the most number of treasures according to μ among the doors not yet guessed; in case of equality, she picks randomly from the doors with the most number of treasures hidden.

For example, suppose that she has picked allocation μ , where doors 1 and 2 each have two treasures, door 3 has none and door 4 has three treasures. In this case, in the first three rounds she would pick door 4, then in rounds 4 and 5 she can pick door 1 (or 2, respectively, with 50% chance) and finally in rounds 6 and 7 door 2 (or 1, respectively). Note that if we were to depict the treasures found at subsequently guessed doors, we would obtain a monotone decreasing sequence, in other words, a Young diagram. We denote the (unique) Young diagram with only one element by “.” in subscript.

Still assuming $k = 1$, suppose that so far the searcher has already found some treasures behind some doors using the above strategy, giving some Young diagram λ . For an outside observer who doesn't know μ , the above rules exactly determine a probability $p_\lambda(n, d, 1)$ with which she would continue guessing her current door, or guess a new door instead with probability $q_\lambda(n, d, 1) = 1 - p_\lambda(n, d, 1)$, selected uniformly from the so far unguessed doors. For example, if $n = 2$ and $d = 2$, and in the first round she has found a treasure behind door 1, then she would again guess door 1 with probability $p_\lambda(2, 2, 1) = \frac{2}{3}$ and guess door 2 with probability $q_\lambda(2, 2, 1) = \frac{1}{3}$; this is because the conditional expectation for μ after the first round is that with probability $\frac{2}{3}$ both treasures are behind door 1 and with probability $\frac{1}{3}$ there is one treasure behind each door (in this case she had 50% chance to start with door 1 and not door 2). More generally, after finding a treasure in the first round, she would guess a new door with probability $q_\lambda(n, d, 1) = \frac{\binom{n}{d}}{\binom{n+d-1}{d}}$, and after this her only option would be to keep on guessing a new door in each round. Note that $p_\lambda(n, d, 1) \rightarrow 0$ as $n \rightarrow \infty$ for any fixed λ .

Now we are ready to present the strategy of the searcher for $k > 1$ for large enough n . It will again be such that at any time the treasures found behind subsequent doors form a Young diagram λ . In each round searcher guesses the last guessed door and guesses $k - 1$ new doors with probability $p_\lambda(n, d, k) = k \cdot p_\lambda(n, d, 1)$, and otherwise guesses k new doors with probability $q_\lambda(n, d, k) = 1 - p_\lambda(n, d, k)$. (Where the new doors are always selected uniformly at random from the so far unguessed doors.) For new doors to be always available, we need that $n \geq dk$ (as conjectured by Cs3ka), but for $p_\lambda(n, d, k)$ to denote a probability (i.e., for $p_\lambda(n, d, k) \leq 1$), it might be needed that n is even larger; e.g., if $n = 6$, $d = 3$ and $k = 2$, we would get $p_\lambda(6, 3, 2) > 1$ - for a detailed discussion of small values of n , see Section 4.

To finish the proof, we need to show that the above strategy indeed finds any allocation μ of the treasures with probability $\frac{k^d}{\binom{n+d-1}{d}}$. Note that if at anytime the guess contains more than one door that hides undiscovered treasures, searcher will eventually lose. Thus, it is enough to show that for each λ the strategy for general k has exactly k times as much chance of guessing any door i in the next round as the strategy for $k = 1$. It is important that here we consider only how this probability depends on λ , i.e., for $k > 1$ the doors that were guessed earlier but were not revealed to hide a treasure are ignored and treated as not yet guessed.

There are two cases to consider. The first case is if door i is where searcher has last found a treasure. In this case searcher's probability of guessing it again is $p_\lambda(n, d, k) = k \cdot p_\lambda(n, d, 1)$. The second case is if door i is a new, yet unguessed door. In this case searcher's probability of guessing door i equals the expected number of newly guessed doors divided by the number of doors that are not yet known to contain a treasure (including the already guessed doors that were not revealed to hide a treasure). The denominator is the same for $k > 1$ and $k = 1$, thus it is enough to consider the nominator. According to our strategy, the expected number of newly guessed doors equals $k - 1 + 1 - k \cdot p_\lambda(n, d, 1) = k \cdot q_\lambda(n, d, 1)$. This finishes the proof. \square

Remark 10. One could also give the equation for each λ that the probabilities need to satisfy so that the probability of finding each λ -shaped treasure allocations is $\frac{k^d}{\binom{n+d-1}{d}}$. For example, the probability of finding a $(2, 1)$ -shaped allocation is $\frac{k}{n} \cdot p(n, 3, k) \cdot \frac{(k-p(n, 3, k))}{n-1} = \frac{k \cdot p(n, 3, k) \cdot (k-p(n, 3, k))}{n(n-1)}$. This way we get $p(d)$ equations, where $p(d)$ denotes the *partition function*, which equals the number of Young diagrams of size d . Note that since the strategy always finds an allocation and no allocation is found in two different ways, one of these equations is redundant. The number of variables ($p_\lambda(n, d, k)$'s) of these equations equals the number of Young diagrams of size at most $d-1$ with the additional property that the smallest part doesn't equal the second smallest part. This number is exactly $p(d) - 1$, the number of equations minus one redundant; this is because we can consider each event associated to $p_\lambda(n, d, k)$ as a choice, and every sequence of choices returns a different Young diagram.* Although we could not prove that there are no additional redundant equalities and the equations are non-linear, this suggests that there is only one choice for the $p_\lambda(n, d, k)$ (that follows our strategy).

4 Small n

As can be seen from the proof of the upper bound of Theorem 1, any optimal searching strategy that attains the upper bound must guess k new doors in each round. This also implies that $v_1(n, d, k) < \frac{k^d}{\binom{n}{d}}$ if $n < dk$.

The situation for the Multiple Caching Game is different.

If $d = 1$, then the bound in Theorem 2 is sharp for every $n \geq k$.

If $d = 2$, the bounds are still always sharp if $n \geq dk$. For $n \leq 2$, this is trivial. For $n \geq 4$, it can be checked that $p_\lambda(n, 2, k) = k \cdot p_\lambda(n, 2, 1) \leq 1$ always holds. For $n = 3$ and $k = 1$ or $k = 3$, the bounds are again trivially sharp. Finally, the only interesting case left is if $n = 3$ and $k = 2$; here we get $p(3, 2, 2) = 2 \cdot p(3, 2, 2) = 1$ and thus $q(3, 2, 2) = 0$ - the only possible value for which the algorithm works, as $p(3, 2, 2) \leq 1$ indeed denotes a probability and $q(3, 2, 2) = 0$ means that we never have to guess the (non-existent) fourth door; a coincidence? (The same thing happens for these values in Alpern's Caching Game, and was already observed by Cs6ka.)

If $d = 3$, the first non-trivial case is $k = 2$ and $n = 3$. The uniform hiding strategy gives $v_M(3, 3, 2) \leq \frac{2^3}{\binom{6}{3}} = 80\%$. This, however, is suboptimal, as hiding all three treasures behind the same door (chosen uniformly at random) gives $v_M(3, 3, 2) \leq \frac{2}{3}$. (It is easy to see that in fact $v_M(3, 3, 2) = \frac{2}{3}$.) In general, hiding all treasures behind the same door gives the following upper bound.

Claim 11. $v_M(n, d, k) \leq \frac{k}{n}$.

It is also interesting to study $d = 3$, $k = 2$, $n = 6$, the first case when $p(n, d, k) > 1$, as $q(6, 3, 1) = \frac{\binom{6}{3}}{\binom{6+3-1}{3}} = \frac{20}{56}$ and $p(6, 3, 2) = 2 \cdot p(6, 3, 1) = 2 \cdot (1 - q(6, 3, 1)) = 2 \cdot \frac{36}{56} > 1$. The uniform hiding strategy gives $v_M(6, 3, 2) \leq \frac{2^3}{\binom{6+3-1}{3}} = \frac{1}{7}$, which is better than the bound of Claim 11. And indeed, $v_M(6, 3, 2) = \frac{1}{7}$ can be achieved by the following searcher strategy. In the first round, guess two doors, say 1 and 2. Unless searcher loses immediately, one of them, say 1, is

*I am thankful to P6ter Cs6kv6ri for proving this identity using generating functions before I've realized this simple equivalence.

revealed to have a treasure. In the second round with probability p , searcher guesses doors 1 and 3, and with probability $1 - p$, doors 3 and 4. If she again finds a treasure behind door 1, then in round 3 she guesses doors 1 and 5 with probability p , and doors 5 and 6 otherwise. If she has found the second treasure behind some other door, say, door 3, then in round 3 she guesses doors 3 and 5 with probability p , and doors 5 and 6 otherwise. A simple calculation gives that $p \cdot p = \frac{3}{7}$ and $(2 - p)(2 - p) = \frac{10}{7}$ are the only conditions these probabilities need to satisfy to find any treasure allocation with probability $\frac{1}{7}$. These probabilities are between 0 and 1 if and only if $\frac{4}{7} \leq p \leq 1$.

Note that if $p = 1$, we never need to guess all six doors. This suggests that the upper bound might be again sharp for $d = 3$, $k = 2$, $n = dk - 1 = 5$, just like for $d = 2$. And indeed, the choice of $p = 1$, $p = \frac{4}{7}$ and $p = \frac{6}{7}$ confirms $v_M(5, 3, 2) \leq \frac{2^3}{(5+3-1)} = \frac{8}{35}$. In fact, for any $d = 3$, $n = 3k - 1$ values the choice of $p = 1$, $p = \frac{nk^2}{(n+2)} \rightarrow \frac{2}{3}$ and $p = \frac{n+1}{n+2} \rightarrow 1$ works.

For $d = 3$, $k = 2$, $n = 4$, the uniform hiding strategy still gives a better bound than Claim 11, but apparently it cannot be attained, as “there are not enough doors.” This seems like a good candidate where the different variants of the game might have different values.

5 Concluding remarks

What happens if n is small? Do some $p_\lambda(n, d, k)$ probabilities exist for every $n \geq dk$? A possible approach might be to prove their existence using some argument similar to the one used Baranyai’s theorem [3]. Can it ever be useful to pick less than k doors? We could not eliminate this possibility for $n < dk$. Does it matter whether get a random treasure or hider reveals one? What if instead a beneficent helper of searcher reveals the treasures? Does $v_A(n, d, k) = v_M(n, d, k)$ always hold?

What happens if we need to find only d' of the d treasures?

What happens if we can make e unsuccessful guesses?

What happens if instead of bounded query size we impose other restrictions on the queries in caching games of this type?

Acknowledgment

I am thankful to Endre Csóka for discussions on the topic.

References

- [1] S. Alpern, R. Fokkink, T. Lidbetter, and N. S. Clayton, A search game model of the scatter hoarder’s problem, *Journal of The Royal Society Interface* 9(70):869–879, 2012.
- [2] Search theory: a game theoretic perspective (S. Alpern, R. Fokkink, G. Leszek, R. Lindelauf, V. S. Subrahmanian (Eds.)), Springer Science & Business Media, 2013.
- [3] Zs. Baranyai, On the factorization of the complete uniform hypergraph, in *Infinite and Finite Sets* (A. Hajnal, R. Rado, V. T. Sós, (Eds.)), *Proc. Coll. Keszthely, Colloquia Math. Soc. János Bolyai* 10, North-Holland, 91–107, 1973.
- [4] E. Csóka, Limit theory of discrete mathematics problems, preprint, [arxiv.org/1505.06984v2](https://arxiv.org/abs/1505.06984v2).
- [5] E. Csóka and T. Lidbetter, The solution to an open problem for a caching game, *Naval Research Logistics* 63(1):23–31, 2016.